Live TV Station Broadcasting by Utilizing Windows Server2008 (Windows Media Service) and Video Advertisement Management by Utilizing Server-side Playlist programming

Ding Luo

STD#: 728355

Dec. 2008

Abstract

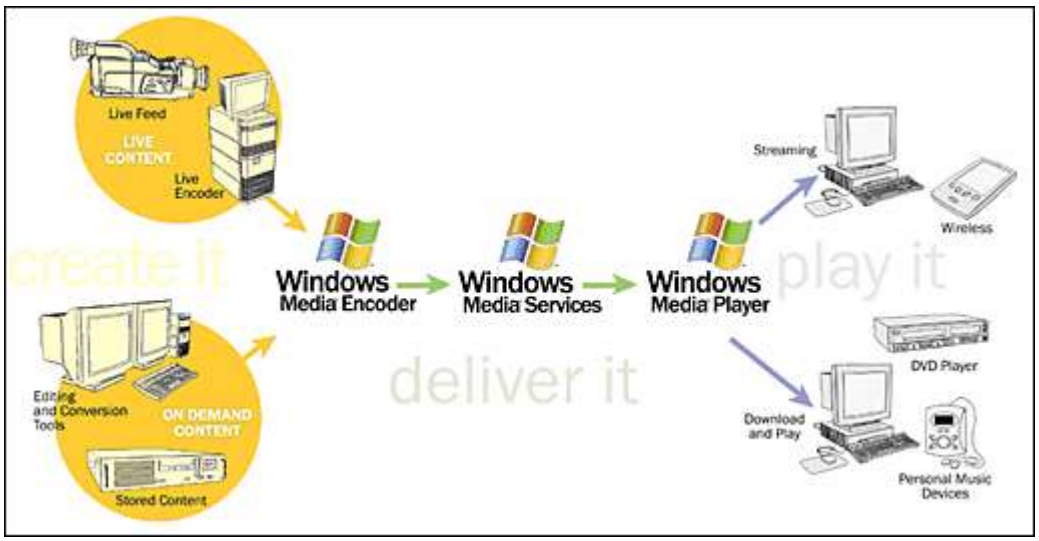
This project has two parts: one is going to setup a live TV video station utilizing Windows Media Services on Windows Server 2008. This video station can transfer video clip by using on-demand and broadcast two types with or without advertisement, and streaming videos programs through 5 channels, which are : MTV, MVE, Silverlight, CS440, and Air Force; another is an advertisement management web application. The main function of this web application includes customer management, ad upload and management, playlist programming and scheduling which based on SQL server. A sample web application was built to utilize the advertisement management at http://dingluolivevideo.no-ip.biz:8000/ .

Contents

Part I Internet Live TV Station

Introduction

Internet Live TV Station utilized windows media services to broadcast videos on internet. Users can have the same experience as they watch real TV. They click the stream, and it starts playing instantly. There are no more customer service calls because of lengthy buffering! Users can switch back and forth between streams—from golf to baseball to basketball to soccer—with remote-control-like swiftness. And, with new "Instant-On/Always-On" capabilities, viewers can now watch content on the Web just like they watch it on TV, because the "Fast Cache" feature caches enough of the streaming content to ensure that it doesn't start to re-buffer in the middle of the content or get disconnected. The business logic of Internet Live TV Station includes 2 parts: server side and client side. Server side: video sources, Windows Media Encoder, Windows Media Services; client side: Widows Media Player.



Internet Live TV Logic Diagram

Because the Windows Media Services can only streaming Windows Media Video (WMV) format, which is a compressed video file format for several proprietary codecs developed by Microsoft for Internet streaming applications. A Windows Media Encoder is needed to encode video files to WMV format and also to stream live content.

Windows Media Services is a platform for streaming TV video content to clients over the Internet or an intranet. These clients may be other computers or devices that play back the content using a player, such as Windows Media Player, or they may be other computers running Windows Media Services (called Windows Media servers) that proxy, cache, or redistribute your content. Clients can also be custom applications that have been developed using Windows Media SDK Components.

The content that Windows Media server streams to clients here is a live stream. You can also stream files that have been encoded by Windows Media Encoder, Microsoft Producer for Microsoft Office PowerPoint 2003, Windows Movie Maker, and many third-party encoding programs.

You can use Windows Media Services to configure and manage one or more Windows Media servers that deliver your content to clients. Simplified deployment and administration ensures that you can easily set up and administer Windows Media servers, even if you are not familiar with streaming concepts.

Windows Media Services can only run on windows server system: 2000/2003/2008. It has to setup a windows server system for utilizing windows media services to streaming videos. This project is based on Windows Server 2008 with Windows Media Services 2008

Windows Server 2008 Installation and Configuration

Windows Server 2008 Installation

There has a windows server 2008 for free download from Microsoft's Windows 2008 Server Trial website. After download, you can install Windows Server 2008 by inserting your Windows Server 2008 installation media into your DVD drive. Reboot computer you will get the prompt for installation. Follow the installation instruction and choose appropriate options, then click next to finish this installation procedure. Then the server reboots you'll be prompted with the new Windows Server 2008 type of login screen. Press CTRL+ALT+DEL to log in. You may be asked to change your password when you logged in Windows Server 2008 the first time.

After you logged in Windows Server 2008 as a administrator, you have to configure this server in order to activate its function

Windows Server 2008 Configuration

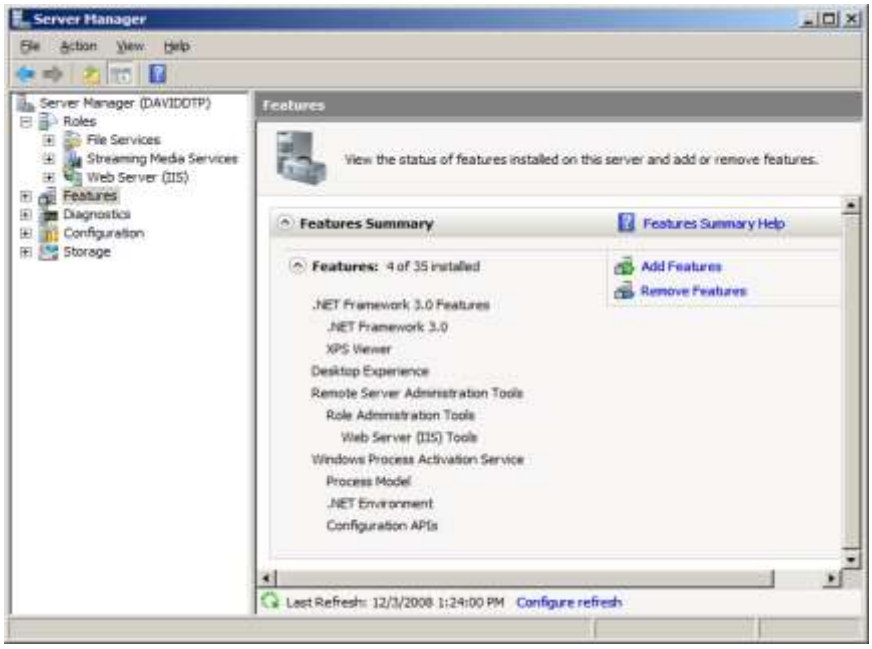
This server plays as a development platform. The VS 2008, Windows Media Services, Windows media player, Windows Media Encoder, WebCam etc., all will run on it. If this server configured as a desktop will be very easy for developing applications.

Activation of Windows Server 2008 Desktop Feature

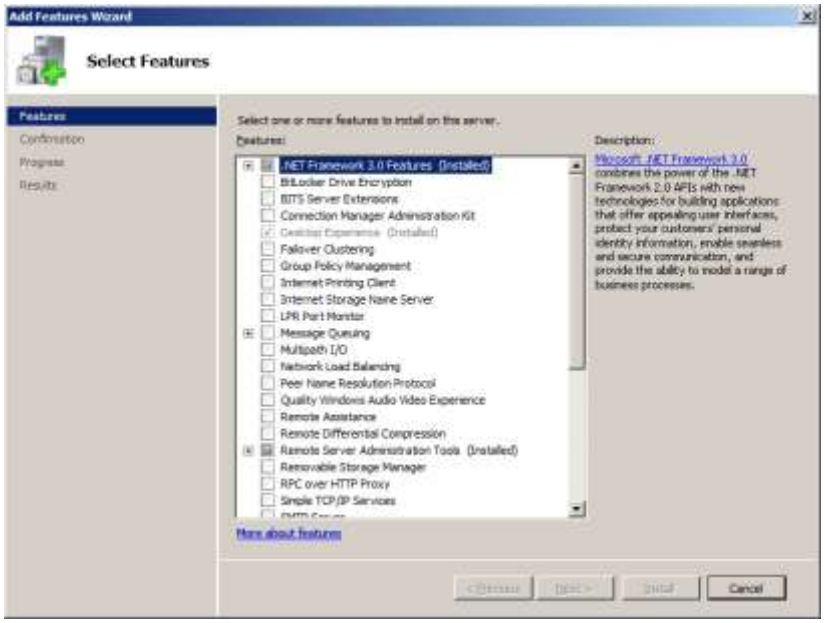
Open Server Manager

click Features on left pane

click Add Feature in the right pane



check desktop experience in feature list



click install and finish

Tips: You can treat Windows Server 2008 as Windows Vista, so you can find device drive for your computer such as sound card, monitor drive, webcam etc.

Windows Media Services Installation and Configuration

Windows Media Services Installation

Windows Media Services, which is the most important component in our Internet Live TV system. It is not shipped with Windows Server 2008. You can download Windows Media Services 2008 at http://www.iis.net/downloads/default.aspx?tabid=34&i=1612&g=6

Install Windows Media Services by following the instruction. This should no any problem. You cannot see the Windows Media Service Consol in Windows Server 2008 until you add it as a role in Windows Server 2008.

Open Server Manager in Windows Server 2008

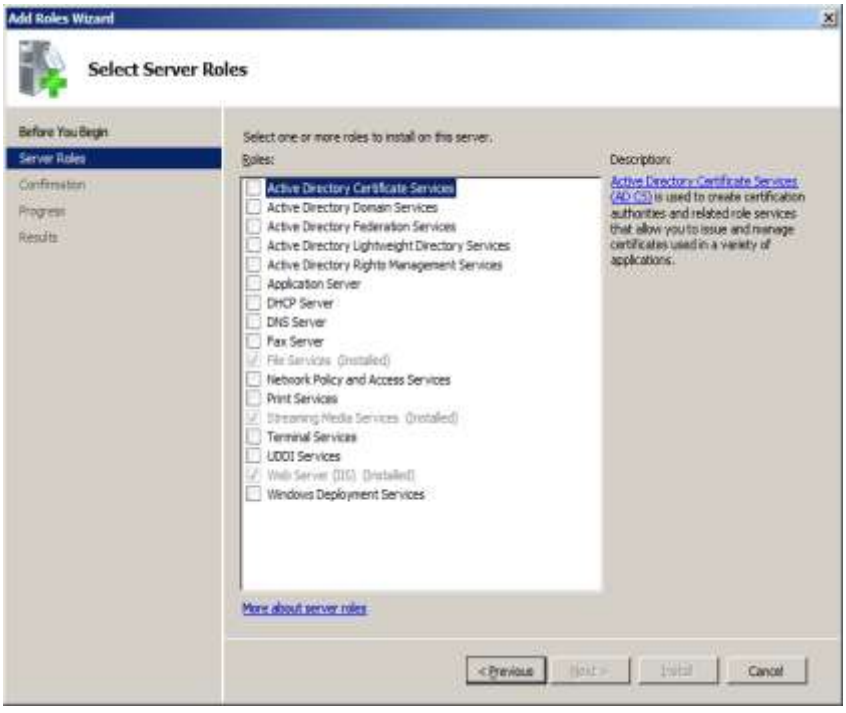
Click Roles in left pane

Click Add Roles in right pane

Click Server Roles in Add Roles Wizard

Check Streaming Media Services

Click install and next to finish



Now the Windows Media Services Console is appeared in Windows Server 2008. Next step is to configure the Windows Media Services to distribute those video contents.

Windows Media Services Configuration

The basic steps involved in setting up a Windows Media server include adding and configuring publishing points to identify the content you plan to stream, and communicating to users that the content is available.

A Windows Media server uses publishing points to translate a client request for content into a physical path on the server hosting the content. You can add two types of publishing points to a Windows Media server, broadcast and on-demand. If you want to stream live content from an encoder, a broadcast publishing point is the best choice. If you plan to stream a file and want to allow users to control playback of the content (for example, to pause, rewind, or fast-forward it), an on-demand publishing point is the best choice.

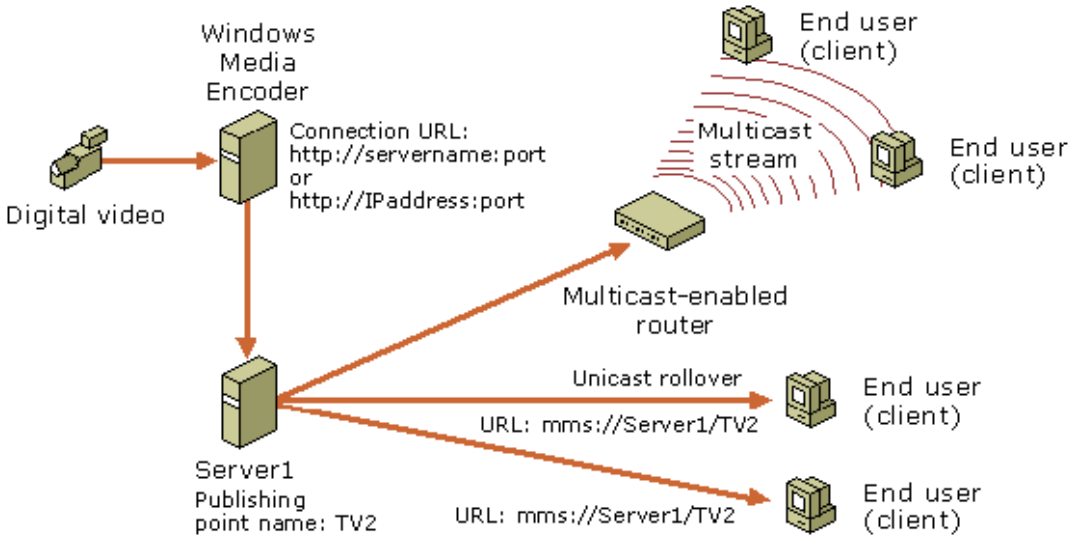
After you have added a publishing point and identified the content you want to stream from it, you need to communicate that the content is available. An easy method for accomplishing this is to create an announcement for the content.

You may also want to implement some of the more advanced features available through Windows Media Services. For example, you can modify settings to limit the number of client connections, set up security measures to protect your content, log data about client activity, and set up a distribution server.

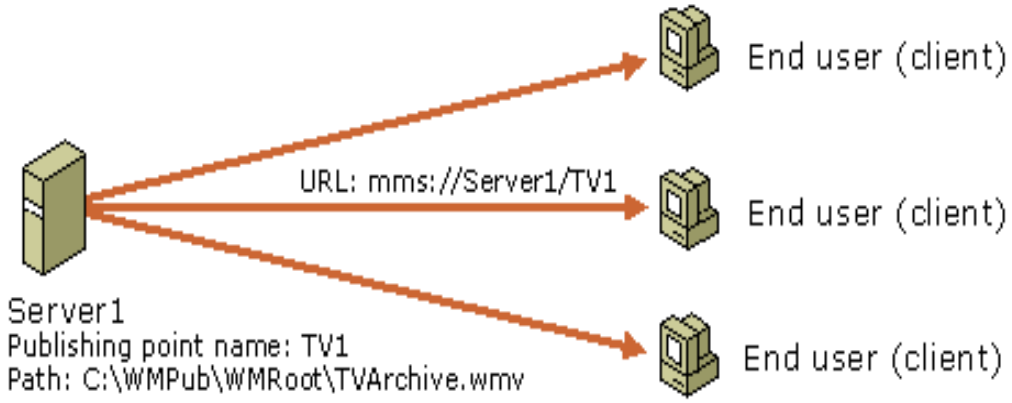
When selecting the type of publishing point to use, you should consider how you want to deliver the content; for example, whether you want to deliver the content as a unicast or multicast stream. With a unicast stream, clients connect to a Windows Media server to access content.

With a multicast stream, the server streams content to a single multicast IP address on the network, and all clients access that IP address to receive the stream instead of connecting to the server. This

reduces the amount of bandwidth required on the network as the single stream is able to fulfill multiple client requests.



You can deliver content as a unicast stream from either an on-demand or a broadcast publishing point. A unicast stream is a one-to-one connection between the server and a client, which means that each client receives a distinct stream and only those clients that request the stream receive it. Unicast streaming is the default method by which a Windows Media server delivers content. Unicast streaming is automatically enabled by the WMS Unicast Data Writer plug-in, which is enabled by default

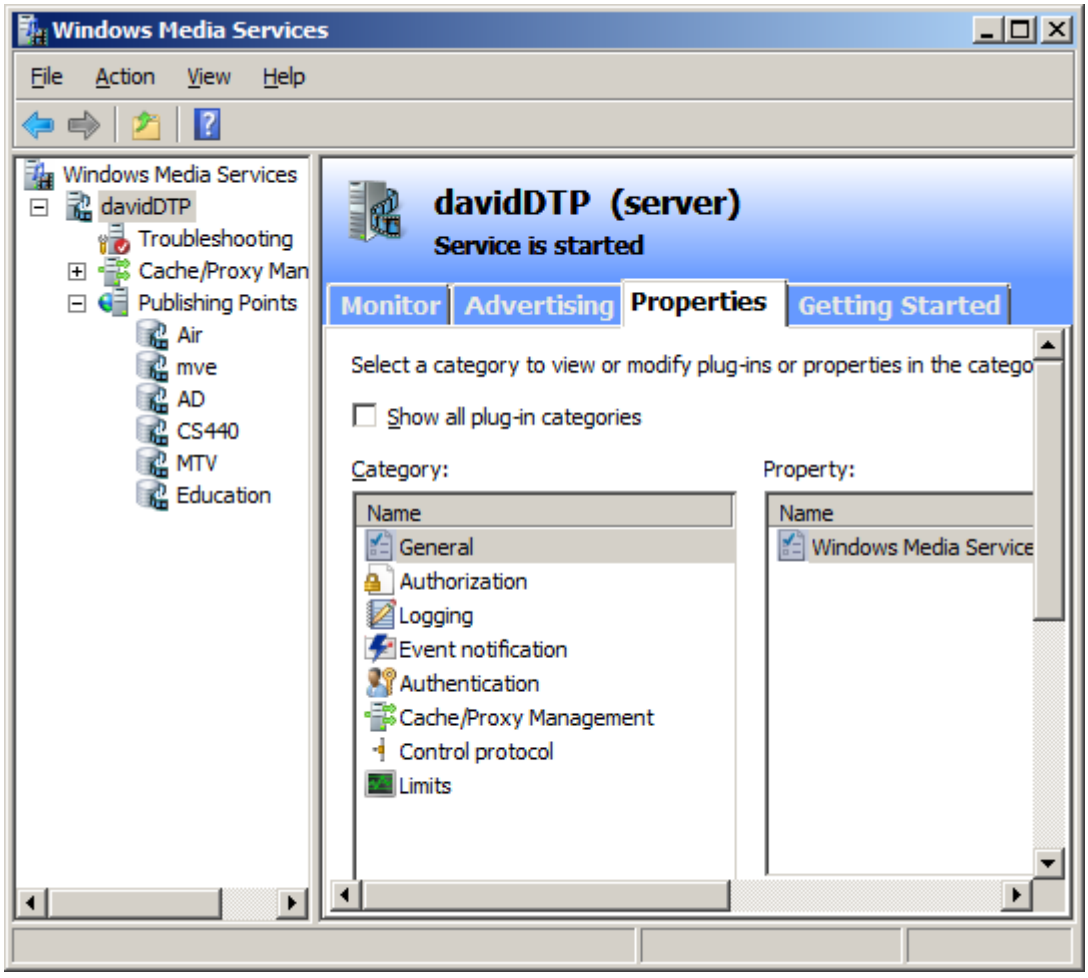


Configuration Steps

Start -> Administrator Tools -> Windows Media Service -> Click

This will fire windows media console

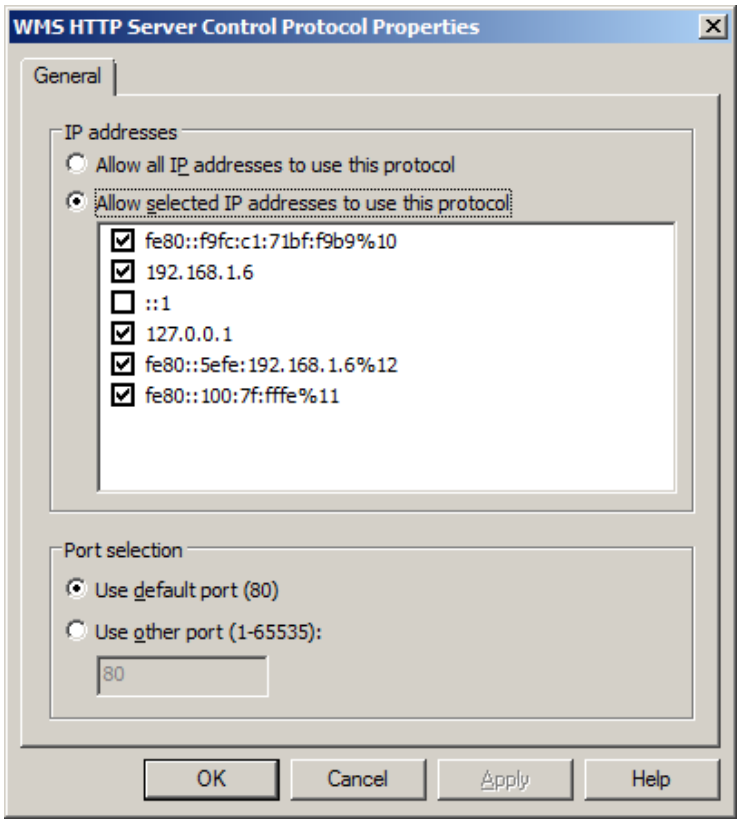
Click Action -> Add Server to fire add server wizard. This will help you add a video server on Windows Server 2008



Click Video Server Node (davidDTP)

Click Properties on the right pane to configure this video server such as authorization, authentication, control protocol etc. You can change the server ip address by edit control protocol to avoid the ip confliction with other services

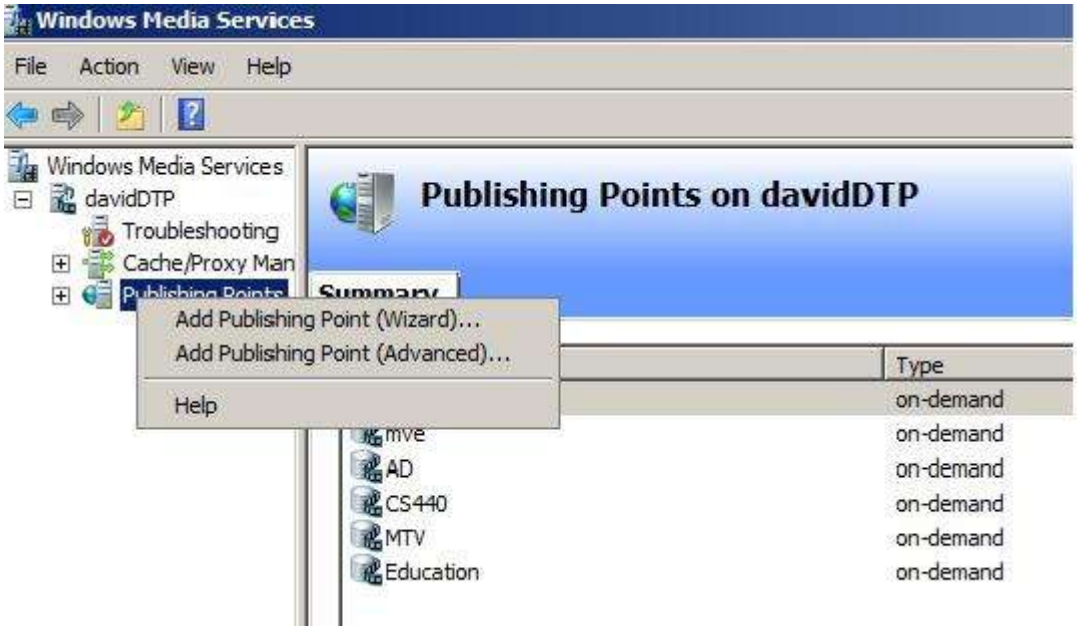
The following picture presents the IP address and port number settings.



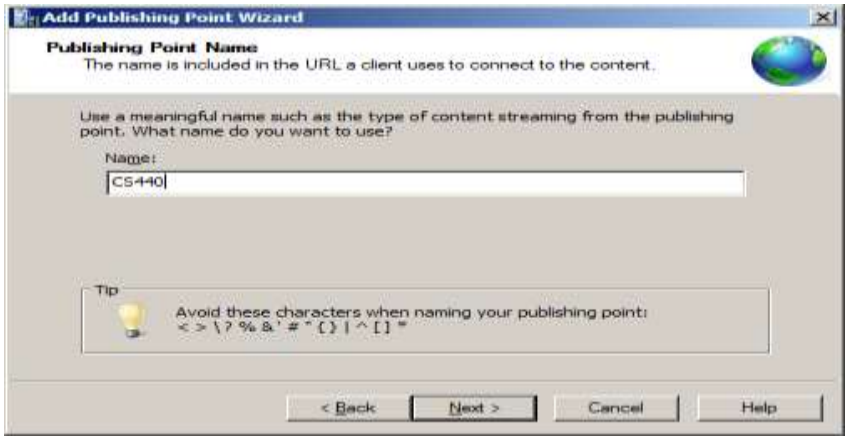
.

Example of Adding a publish point into Windows MediaServices

To add a publish point to this server by right clicking publish points node, then Add publish point and click.



This will fire the publish point wizard. Following the instruction to finish adding a publish point



Part II Advertisement Schedule Web Application

Introduction

Advertisement is now playing an important role in current commercial website. It is obviously that a Web Application is needed for customer who want to presents their AD video clip before some movie program. This is one of part of this project: to develop a Web Application which insert customer AD video clip into Internet Live TV Station regular program and deliver to clients. The main function of this Web App includes customer controls, customer AD video management, AD order cart, create playlist according to AD order.

Master page design

Master page design is to construct a page for common parts such as video channel links, introductory pages' links, such as Home, Technology, About etc. this information should be shared for every one. The page designed as following picture.

Main Page Design

This page is a main entry for one who wants doing AD on special movies and for Web Master to manage AD orders. They can login as a member after they registered. The administrator can login as admin. Anonymous can check Website technology, know about the creator, TV schedule etc by click the hyperlinks.

The following is the Main Page Screen Shot

User Page Design

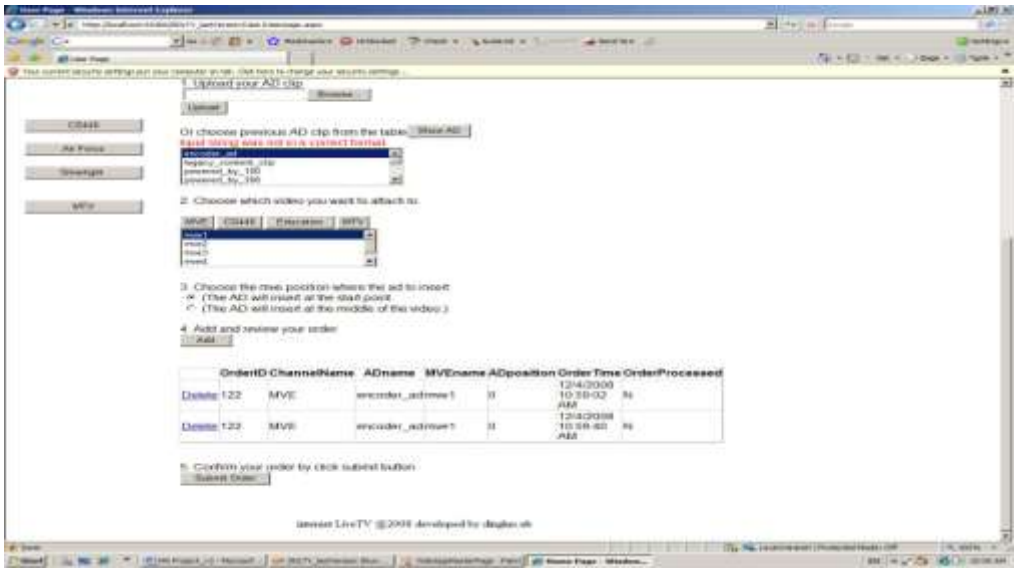
The main functions of this page are:

checking movie video's availability for advertisements

uploading AD video clips

place AD order

AD order Management .



The codes behind the page

Datagridview to explore what movie video is available or not available.

```
protected void mtvx()
    {
        string sqlStr;
        sqlStr = @"select m.mveName as [Video Name] , c.cnlName [Channel Name], a.adName as [AD Name], o.userID
as [Customer Name], m.adAvalible as [AD Available]" +
                  " from mve m " +
                   "left outer join [order] o on m.mveID = o.mveID " +
                   "left outer join [ADtable] a on o.userID = a.userID " +
                   "left outer join [channel] c on m.cnlID = c.cnlID" +
                  " where c.cnlID = 500";//and m.adAvalible is null";
             try
        {
            using (SqlDataAdapter da = new
SqlDataAdapter(sqlStr,ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString))
```

```
            {
                DataSet ds = new DataSet();
                da.Fill(ds, "mveTable");
                gv1.DataSource = ds;
                gv1.DataBind();
            }
        }
        catch (Exception ex)
        {
            infoLabel.Text = ex.Message;
        }
    }
```

Upload AD clip and insert into LTVDB database

```
protected void btnUpload_Click(object sender, EventArgs e)
    {
       if (FileUpload1.HasFile)
        {
            adFileName = FileUpload1.FileName;
            string dirAD = @"C:\inetpub\wwwroot\ADs";//"C:\Documents and Settings\david\My
Documents\Visual Studio 2008\WebSites\DDLTV\ADs\";
           try
            {
                FileUpload1.SaveAs(dirAD + FileUpload1.FileName);
                //  infoLabel.Text = "Your request is under process. Thanks of your business.";
            }
            catch (Exception ex)
            {
                infoLabel.Text = ex.Message.ToString();
            }
            //insert dir and file name to the ad table
            insertAD(dirAD, FileUpload1.FileName);
        }
        else
        {
            infoLabel.Text = "You choose to select AD clip from the previouse ones.";
        }
    }
    protected void insertAD(string dirAD, string adFileName)
    {
        string sqlStr;
        sqlStr = @"select *
                    from [ADtable]
                    order by adID desc";
```

```
            try
        {
    using (SqlDataAdapter da = new
SqlDataAdapter(sqlStr,ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString))
            {
                DataSet ds = new DataSet();
                da.Fill(ds, "ADTable");
                int adID =(int)ds.Tables["ADTable"].Rows[0][0] + 1;
                curUserID = User.Identity.Name;
                DataRow newRow = ds.Tables["ADTable"].NewRow();
                newRow[0] = adID;
                newRow[1] = adFileName;
                newRow[2] = dirAD;
                newRow[3] = curUserID;
                ds.Tables["ADTable"].Rows.Add(newRow);
                SqlConnection cnn = new
SqlConnection(ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString);
                string insertStr = @"insert into [ADtable](adID, adName, adPath,userID)
                    values('" + adID + "','" + adFileName + "','" + dirAD + "','" +curUserID+"')";
                da.InsertCommand = cnn.CreateCommand();
                da.InsertCommand.CommandText = insertStr;
                da.Update(ds,"ADTable");
                infoLabel.Text = "Your Ad clip insert successfully";
            }
        }
        catch (Exception ex)
        {
            infoLabel.Text = ex.Message;
        }
    }
```

Add AD order to shopping cart

```
protected void btnAdd_Click(object sender, EventArgs e)
    {
        if (lbxAD.SelectedValue == "" || lbxMVE.SelectedValue == "")
        {
            lblReview.Text = "You are not chosen anything, choose items from mve table and ad table then
click add to review";
        return;
        }
        if (Session["crtTable"] == null)
        {
            addCart();
```

```
        }
        else
        {
            DataTable dt = new DataTable();
            dt = (DataTable)Session["crtTable"];
            DataRow crtRow = dt.NewRow();
            crtRow["OrderID"] = (int)Session["orderID"];//curOrderID;
            crtRow["ChannelName"] = (string)Session["ChannelName"];
            crtRow["ADname"] = lbxAD.SelectedItem.ToString();
            crtRow["MVEname"] = lbxMVE.SelectedItem.ToString();
            crtRow["ADposition"] = adPosition.ToString();
            crtRow["OrderTime"] = DateTime.Now;
            crtRow["OrderProcessed"] = "N";
                    dt.Rows.Add(crtRow);
            crtGV.DataSource = dt;
            crtGV.DataBind();
            Session["crtTable"] = dt;
        }
    }
    protected void addCart()
    {
        DataTable cartTable = new DataTable();
        cartTable.Columns.Add(new DataColumn("OrderID", typeof(int)));
        cartTable.Columns.Add(new DataColumn("ChannelName", typeof(string)));
        cartTable.Columns.Add(new DataColumn("ADname", typeof(string)));
        cartTable.Columns.Add(new DataColumn("MVEname", typeof(string)));
        cartTable.Columns.Add(new DataColumn("ADposition", typeof(int)));
        cartTable.Columns.Add(new DataColumn("OrderTime", typeof(DateTime)));
        cartTable.Columns.Add(new DataColumn("OrderProcessed", typeof(string)));
        //cartTable.Columns.Add(new DataColumn("MovieAvalible", typeof(string)));
        string sqlStr = @"select orderID from [order] order by orderID desc";
        SqlConnection cnn = new
SqlConnection(ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString);
        SqlDataAdapter da = new SqlDataAdapter(sqlStr, cnn);
        SqlCommand cmd = new SqlCommand(sqlStr, cnn);
        cnn.Open();
        object odrID = (object)cmd.ExecuteScalar();
        cnn.Close();
        if (odrID == null)
        {
            curOrderID = 100;
            Session["orderID"] = curOrderID;
        }
        else
```

```
        {
            curOrderID = (int)odrID + 1;
            Session["orderID"] = curOrderID;
        }

        DataRow crtRow = cartTable.NewRow();
        crtRow["OrderID"] = curOrderID;
        crtRow["ChannelName"] = (string)Session["ChannelName"];
        crtRow["ADname"] = lbxAD.SelectedItem.ToString();
        crtRow["MVEname"] = lbxMVE.SelectedItem.ToString();
        crtRow["ADposition"] = adPosition.ToString();
        crtRow["OrderTime"] = DateTime.Now;
        crtRow["OrderProcessed"] = "N";
        cartTable.Rows.Add(crtRow);
        crtGV.DataSource = cartTable;
        crtGV.DataBind();
        Session["crtTable"] = cartTable;
        }
```

Submit Order

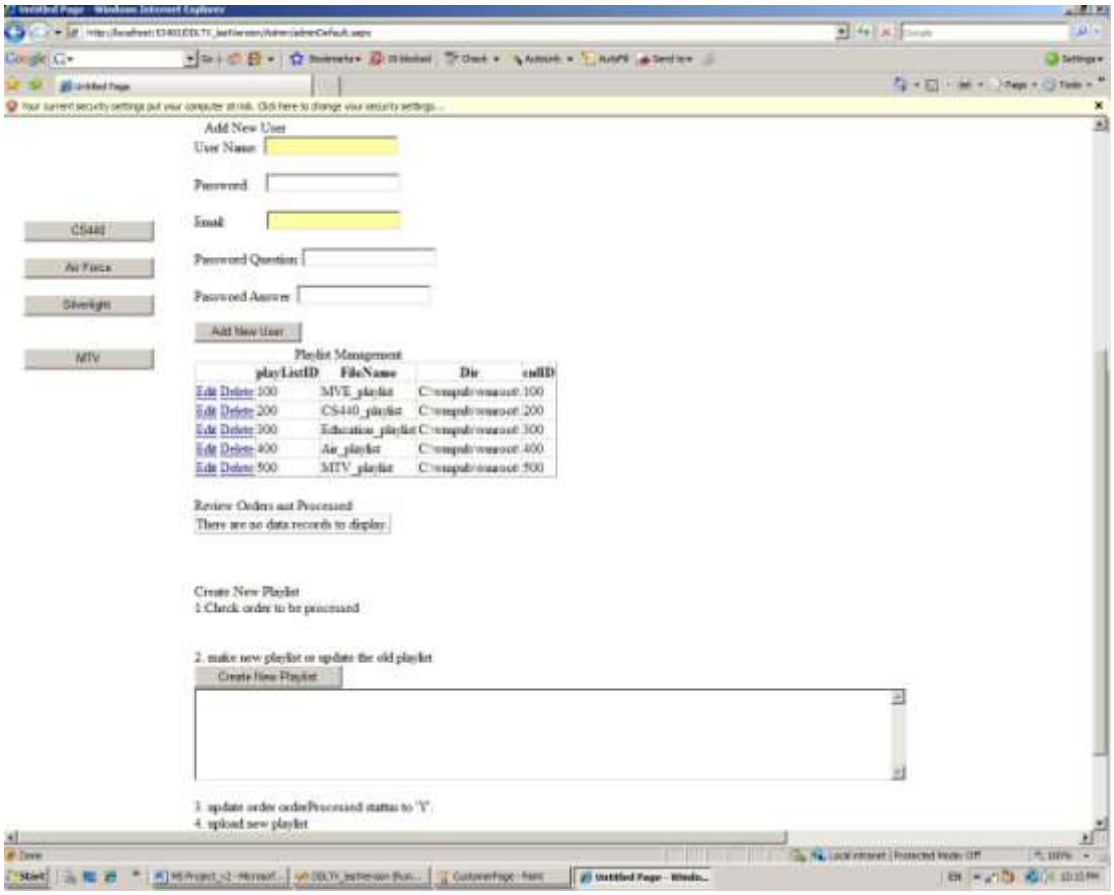
```
protected void btnSbt_Click(object sender, EventArgs e)
    {
        int _oID, _adID, _mveID, _cnlID;
        string _uID;
        DateTime  _tID;
        string _oFlgID;
        DataTable tmp = new DataTable();
        tmp = (DataTable)Session["crtTable"];
        //orderID, adID, mveID, cnlID, userID, orderTime
        string cnnStr =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
       // SqlDataAdapter sqlDa = new SqlDataAdapter();
        SqlConnection cnn = new SqlConnection(cnnStr);
        SqlCommand cmd = new SqlCommand();
        string commandText = "insert into [order] values(@oID,@aID,@mID,@cID,@uID,@tID,@fID)";
            try
            {

                foreach (DataRow dr in tmp.Rows)
                {
                    cmd = new SqlCommand(commandText, cnn);
                    cnn.Open();
```

```
            _oID = (int)dr["OrderID"];
            _adID = getID(dr["ADname"].ToString(), 1);
            _mveID = getID(dr["MVEname"].ToString(), 2);
            _cnlID = getID(dr["ChannelName"].ToString(), 3);
            _uID = User.Identity.Name;
            _tID = (DateTime)dr["OrderTime"];
            _oFlgID = (string)dr["OrderProcessed"];
            cmd.Parameters.Add("@oID", SqlDbType.Int).Value = _oID;
            cmd.Parameters.Add("@aID", SqlDbType.Int).Value = _adID;
            cmd.Parameters.Add("@mID", SqlDbType.Int).Value = _mveID;
            cmd.Parameters.Add("@cID", SqlDbType.Int).Value = _cnlID;
            cmd.Parameters.Add("@uID", SqlDbType.VarChar).Value = _uID;
            cmd.Parameters.Add("@tID", SqlDbType.DateTime).Value = _tID;
            cmd.Parameters.Add("@fID", SqlDbType.Char).Value = _oFlgID;
            cmd.ExecuteNonQuery();
            cnn.Close();
        }
        lblSubmitInfo.Text = "Your order submitted successfully. Thank you";
    }
    catch (Exception ex)
    {
        lblSubmitInfo.Text = ex.Message;
    }
    cmd = new SqlCommand(commandText, cnn);
commandText = "update mve set adAvalible = @advbl where mveID = @mID";
    try
    {
       foreach (DataRow dr in tmp.Rows)
        {
            cmd = new SqlCommand(commandText, cnn);
            cnn.Open();
            _mveID = getID(dr["MVEname"].ToString(), 2);

            cmd.Parameters.Add("@advbl", SqlDbType.VarChar).Value = 'N';
            cmd.Parameters.Add("@mID", SqlDbType.Int).Value = _mveID;

            cmd.ExecuteNonQuery();
            cnn.Close();
        }
        lblSubmitInfo.Text = "Your order submitted successfully. Thank you";

    }
    catch (Exception ex)
    {
```

```
                lblSubmitInfo.Text = ex.Message;
            }
    }
    protected int getID(string idName, int tblFlag)
    {
        int rtnVal;
        string cnnStr =
ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;

       // SqlDataAdapter sqlDa = new SqlDataAdapter();
        SqlConnection cnn = new SqlConnection(cnnStr);
        string commandText = "";
        switch(tblFlag)
        {
            case 1:
                commandText = "select adID from [ADtable] where adName = @aName";
                break;
            case 2:
                commandText = "select mveID from [mve] where mveName = @aName";
                break;
            case 3:
                commandText = "select cnlID from [channel] where cnlName = @aName";
                break;
        }
        SqlCommand cmd = new SqlCommand(commandText, cnn);
        cmd.Parameters.AddWithValue("@aName", idName);
        cnn.Open();
        rtnVal = (int)cmd.ExecuteScalar();
        cnn.Close();
        return rtnVal;
    }
}
```

Administrator Page Design

This page is for web master to manage customers, video clips, ad orders and create TV broadcasting playlist



Main Code behind

To add AD video into a specific movie, we have to load the existing playlist enumerate the video node comparing with customer orders and find the AD clip which append to. Then make a new node using this AD clip and insert it before the movie video, the last is save the modified playlist back to Windows Server 2008 for Windows Media Services to streaming out.

```
protected void btnNewPlaylist_Click(object sender, EventArgs e)
    {
        string sqlStr = "Select * From [order] where orderProcessed = 'N'" ;
        string cnnStr = ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString;
        SqlDataAdapter da = new SqlDataAdapter (sqlStr,cnnStr);
        DataTable dt = new DataTable();
```

```
        da.Fill(dt);
        bool pflg = true;
        Playlist playList = new Playlist();
        string lclTableName = "";
        ArrayList cnlList = new ArrayList();
        CNL _cnl = new CNL();
        foreach (DataRow _odr in dt.Rows)
        {
            MVE_AD mad = new MVE_AD();
           _cnl.ClName = Order.get_idName_order((int)_odr[3], "channel");
            if(cnlList.Count==0)
            {
                mad.MvName = Order.get_idName_order((int)_odr[2], "mve");
                mad.AName = Order.get_idName_order((int)_odr[2], "AD");
                _cnl.mADs.Add(mad);
                cnlList.Add(_cnl);
                lbl_adminDefault.Text += mad.AName;
            }
            else
            {
                //foreach (Playlist plist in playLists)
                foreach (CNL cl in cnlList)
                {
                    if (cl.ClName == _cnl.ClName)
                    {
                        mad.MvName = Order.get_idName_order((int)_odr[2], "mve");
                         mad.AName = Order.get_idName_order((int)_odr[2], "AD");
                         _cnl.mADs.Add(mad);
                         pflg = false;
                    }
                }
                if (pflg)
                {
                    CNL _cl = new CNL();
                    mad.MvName = Order.get_idName_order((int)_odr[2], "mve");
                    mad.AName = Order.get_idName_order((int)_odr[2], "AD");
                    _cl.mADs.Add(mad);
                    cnlList.Add(_cl);
                    pflg = true;
                }
            }
           }
     modifyThisPlaylist(cnlList);
    }
```

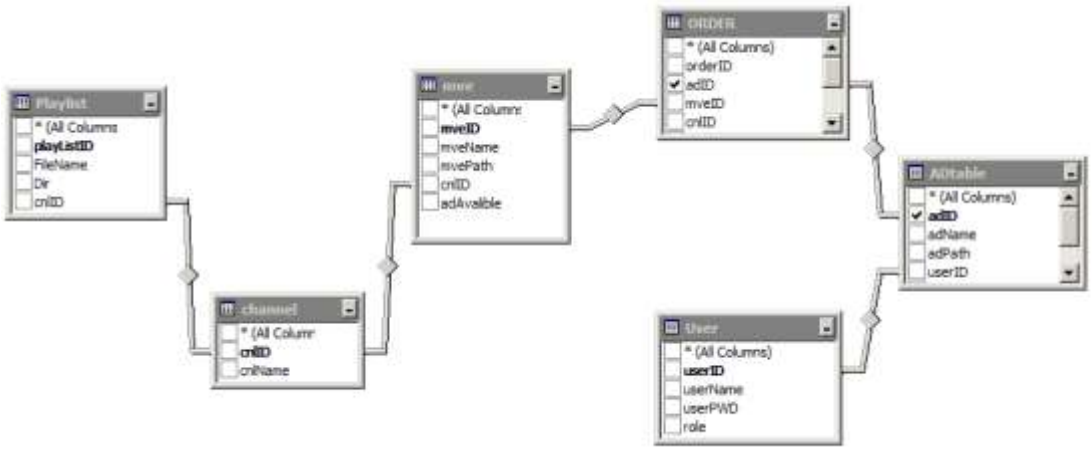
```
    private void  modifyThisPlaylist(ArrayList cList)
    {
        WMSServer Server;
        IXMLDOMDocument plylist;
        IXMLDOMElement ElementMedia;
        IXMLDOMElement[] Ad_Elements = new IXMLDOMElement[256];
        IXMLDOMElement Ad_Element;
        IXMLDOMElement Root_Node;
        IXMLDOMElement New_Element;
        IXMLDOMNodeList NodeList;
        IXMLDOMNode Attr;
        IXMLDOMNode currNode;
        IXMLDOMNode Next_Child;
        int iIndex;
        long lSongCounter;
        long lAdNumber;
        int midP = 0;
        int oid = 0;
        int lclCount = 0;
        int i;
        string strPath = "";
        Server = new WMSServerClass(); ;
        plylist = Server.CreatePlaylist();
        foreach (CNL _cl in cList)
        {
            try
            {
                strPath = "c:\\wmpub\\wmroot\\" + _cl.ClName.Trim() + "_playlist.wsx";
                lbl_adminDefault.Text += strPath + Environment.NewLine;
                plylist.load(strPath);
                NodeList = plylist.getElementsByTagName("media");
                Root_Node = plylist.documentElement;
                for (i = 0; i < NodeList.length; i++)
                {
                    // Retrieve the next node in the list.
                    ElementMedia = (IXMLDOMElement)NodeList[i];
                    Attr = NodeList[i].attributes.getNamedItem("src");
                   string fullName = Attr.nodeValue.ToString();
                   string mveName = fullName.Substring(fullName.LastIndexOf("\\")+1,(fullName.Length -
fullName.LastIndexOf("\\")-5));
                  foreach( MVE_AD _ma in _cl.mADs)
                   {
                       string lclMvName = _ma.MvName.Trim();
                       string lcllstmveName = mveName.Trim();
```

```
                        if( lcllstmveName == lclMvName )
                        {
                            Ad_Element = plylist.createElement("media");
                            Ad_Element.setAttribute("role", "Advertisement");
                            Ad_Element.setAttribute("src", "c:\\wmpub\\wmroot\\" + _ma.AName + ".wmv");
                            Ad_Element.setAttribute("id", _ma.AName.ToString());
                            currNode = Root_Node.insertBefore(Ad_Element,
Root_Node.childNodes[i+lclCount]);
                            lclCount++;
                            txtPlaylistShow.Text += "c:\\wmpub\\wmroot\\" + _ma.AName + ".wmv" +
Environment.NewLine;
                            txtPlaylistShow.Text += Attr.nodeValue.ToString() + Environment.NewLine;
                            midP = getID(lclMvName, 2);
                            updateMveAdFlag(midP);
                            updateOdrProFlag(midP);
                           }
                    }
                }

            }
            catch (Exception ex)
            {
               txtPlaylistShow.Text = ex.Message;// TODO: Exception handler goes here.
            }
            finally
            {
                plylist.save("c:\\wmpub\\wmroot\\" + _cl.ClName + "_playlist.wsx");
            }
        }
    }
```

Tip: you have add reference to Microsoft.WindowsMediaServices.Interop, system.Runtime.InteropServices, and interop_msxml;

Live TV Database (LTVDB) Design

LTVDB is built on SQL Server 2005 Express. The relationship of the tables diagram is showed below.



Part III Conclusion

This project utilized Windows Media Services on Windows Server 2008 and successfully built an Internet Live TV Station. While as the same time developed a Web Application for TV station to manage their AD orders and their customers. It made a sample resolution for commercializing Internet Live TV Station by using Windows Media Services as streaming platform.. Because this project is more focus on the server side, it leaves a lot of studying place on client side such as utilizing Silverlight Technology to make customers get more rich interactive experience.

References

1. Getting Started with Windows Media Services 9 Series
2. Microsoft Windows SDK 6.1
3. Video.Show 1.0 Developer Overview